Lab 0: Introduction to Software-Defined Radio

1 Overview

1.1 Introduction

Welcome to the 4C21 SDR laboratory! In the first part of this term we will use the SDR laboratory sessions to help you better understand the concepts of SDR and to use them in specific applications in communication systems. The experiments provide illustrations and visualisations of the many mathematical concepts you may have studied previously in Next Generation Networks and Communication Systems modules. You will explore communication signals and investigate end-to-end communication systems through software-level definitions for a hardware platform.

The hardware used in this laboratory is Analog Device's ADALM PLUTO SDR Module. The hardware is capable of wireless communication using radio waves that are common in many applications today - 4G/5G telecom systems, WiFi, Bluetooth, terrestrial television broadcasts, and satellite communication, among many others. We will deploy some of these applications using the Pluto SDR, giving you an understanding of how such systems work.

The intelligence in SDR is enabled through 'software'. There are a variety of frameworks that allow interaction with SDR hardware (like the ADALM PLUTO) including open source ones like GNU Radio. For our labs, we will use MATLAB and Simulink as the framework for interfacing with the PLUTO modules, allowing abstract models of communication systems to be built rather easily.

The radio spectrum is a public resource and is centrally regulated by each country. The Commission for Communications Regulation (ComReg) is the primary authority for regulation in Ireland (counterpart of FCC in the United States). Because we will be working with radio signals, even transmitting some ourselves in the laboratory, it is important that we comply with these regulations. The current spectrum allocation can be seen here.

For our labs, we will use a range of frequencies to receive broadcast signals around us, with the PLUTO device configured as a receiver. In case of transmissions, we will be using the 2.45 GHz ISM band for our experiments.

1.2 Definitions

Below are some of the definitions used throughout 4C21-SDR lab sheets:

- Baseband Signal: A baseband signal is one in which the signal energy is contained in frequencies relatively close to 0 Hz. That is, the signal energy is contained in the range 0 Hz to f_m. The quantity f_m is then typically referred to as the signal bandwidth. Examples include human voice (20Hz to 5kHz), audio/music (20Hz to 20kHz), and digital logic signals used within an embedded computer system (frequency is of which is dependent on the operating hardware).

- Passband Signal: A passband signal is a one in which the signal energy is contained in a band of frequencies centered about some frequency, f_c. That is, the signal energy is contained in the range $f_c - f_m$ to $f_c + f_m$, where $f_c >> f_m$. The signal bandwidth is $2f_m$.

- Message Signal: A message signal is the information-bearing signal. The purpose of a communication system is to send message signals from a transmitter to a receiver through a communication channel. Message signals are typically baseband signals.

- Carrier Signal: A carrier signal is a waveform that is modified in accordance to a message signal. Carrier signals are typically much higher frequency than message signals. We will consider sinusoidal carrier waveforms, denoted by frequency f_c.

- Modulation: Modulation is the process by which a message signal is imparted on some characteristic of the carrier wave. The information-bearing message signal is referred to as the modulating signal and the result of the modulation process is referred to as the modulated signal. Modulation, typically performed at the transmitter, usually results in a passband signal.

- Demodulation: Demodulation is the process by which the message signal is extracted from a modulated signal. Demodulation, typically performed at the receiver, will ideally result in the original baseband message signal.

2 SDR Hardware

The past decade has brought about significant advancements in both SDR hardware and software systems. In particular, low-cost SDR technology is now widely available for academic, commercial, and general consumer communities. For our lab, we are using the ADALM Pluto device; however, the experiments can also be replicated on a variety of other SDR platforms.

The SDR device parameters and specifications below are specific to the ones useful for 4C21 lab:

1. **Frequency Coverage** - range of radio frequencies that can be captured by the device. This is largely dependent on the tunable range of the RF front-end device. We typically use f_c to denote the center frequency the SDR is tuned to.

   For the PLUTO device, the frequency coverage is 325 MHz to 3.8 GHz

[*NOTE*: The MATLAB/Simulink package extends this to a much wider range of 70 MHz to 6 GHz, but is not certified by the device manufacturer].

2. **Instantaneous Bandwidth** - span of frequencies, centered about f_C, that can be captured by the SDR at once. For an instantaneous bandwidth B, an SDR receiver tuned to f_C would capture RF signals in the band f_C – B/2 to f_C + B/2.

   For the PLUTO device, the instantaneous bandwidth is up to 20 MHz.

NOTE: The MATLAB/Simulink extension package for PLUTO is already installed on your lab machines.

2.1 Verify your installation

Connect a PlutoSDR device to your lab machine via the USB cable provided. Windows should recongnise the device attached as the drivers are already installed.

To verify the MATLAB configuration and communication with the PlutoSDR, type `findPlutoRadio` at the MATLAB prompt. After a few moments, you should see a message beginning with

```
RadioID: ’usb:0’
SerialNum: ’104473ae73930001feff39009c682d27b7’
```

Note that the `SerialNum` entry will vary depending on your specific PlutoSDR.

3 Spectrum Analyser

We will now create a spectrum analyser to scan the wireless range around us using the PLUTO hardware as the radio device. We will be using Simulink to create the model.

3.1 Create the Spectrum Analyser Model

I. Open a new Simulink model and create the model shown in Figure 1. The components used are:

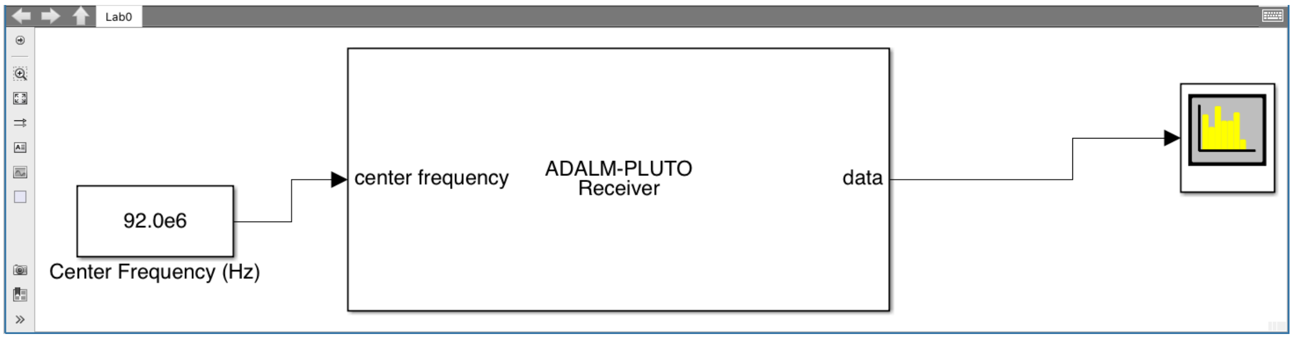


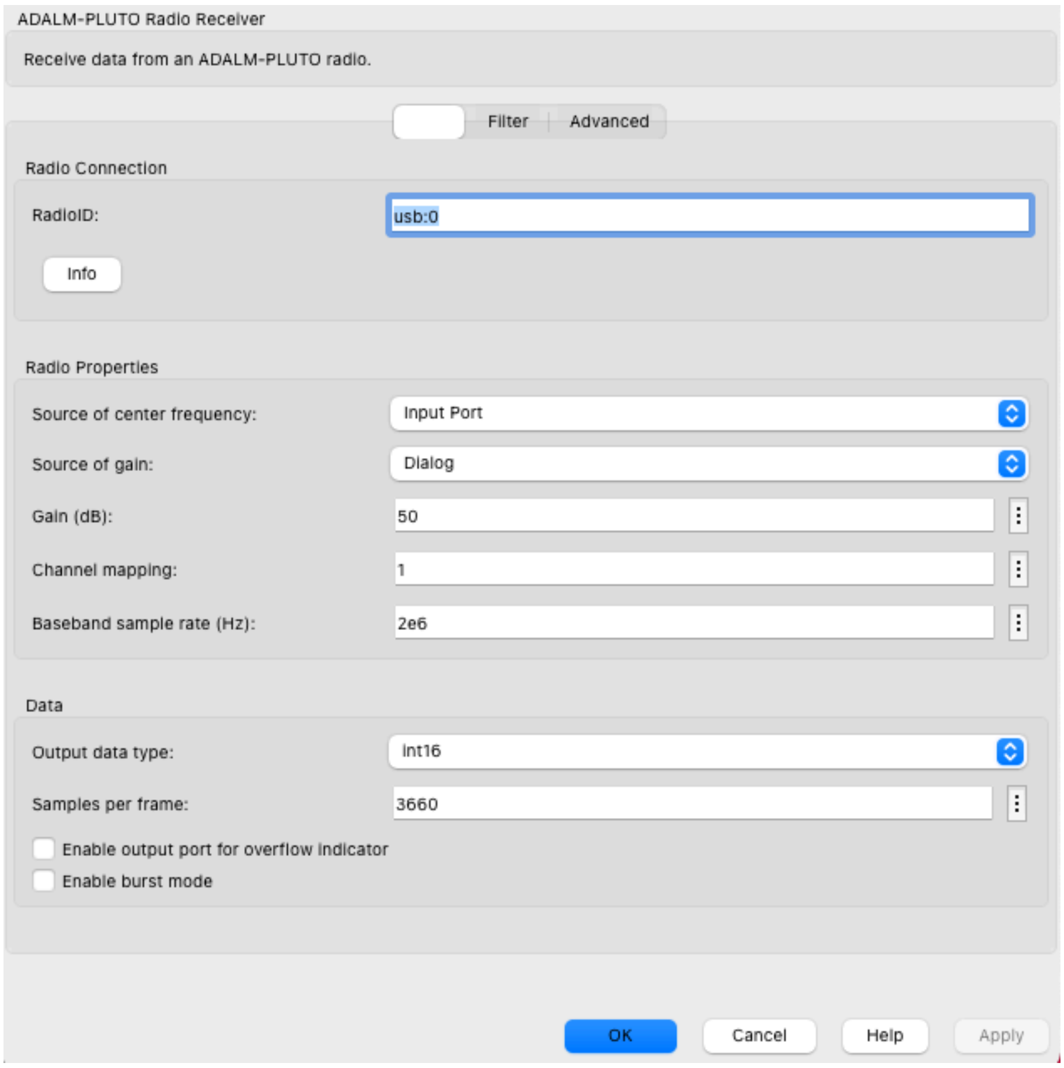
Figure 1: Spectrum Analyser Model

- ADALM-Pluto Radio Receiver (*from* Communications System Toolbox Support Package for ADALM-PLUTO Radio)

- Spectrum Analyser (*from* DSP System Toolbox → Sinks)

- Constant (*from* Simulink → Sources)

II. Set the Simulation stop time to inf.



ADALM-PLUTO Radio Receiver

Receive data from an ADALM-PLUTO radio.

Filter | Advanced

Radio Connection

RadioID: usb:0

Info

Radio Properties

Source of center frequency: Input Port

Source of gain: Dialog

Gain (dB): 50

Channel mapping: 1

Baseband sample rate (Hz): 2e6

Data

Output data type: int16

Samples per frame: 3660

☐ Enable output port for overflow indicator

☐ Enable burst mode

OK | Cancel | Help | Apply

Figure 2: Configuration of the Pluto Receiver Model

III. Configure the ADALM-PLUTO Radio Receiver block as shown in Figure 2. The key settings and their meaning are as follows:

i. Center frequency (Hz) / Source of center frequency: controls the center frequency of the SDR hardware. Selecting Input Port allows the setting to be easily adjusted while the Simulink model is executing. Set the corresponding Constant block to 92.0e6.

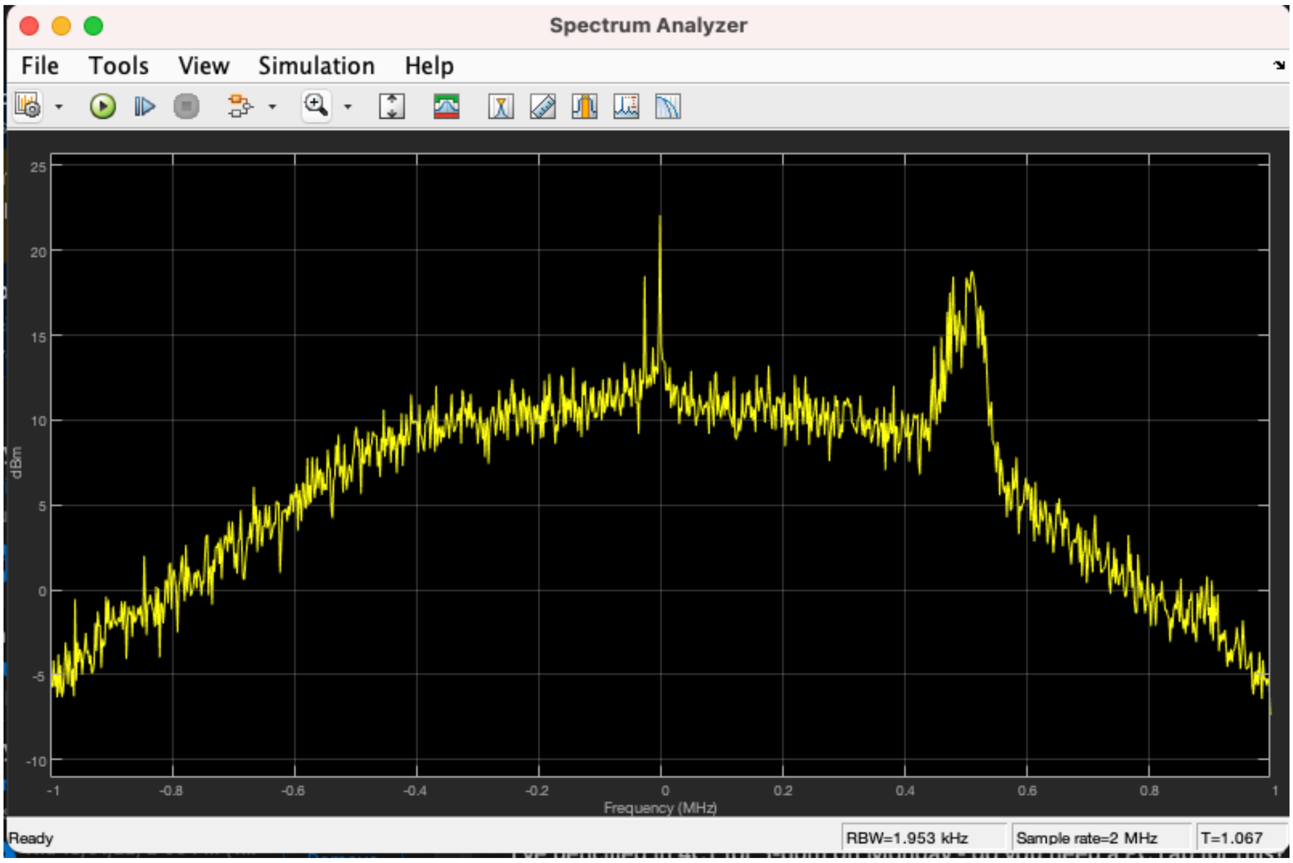
ii. Gain (dB): controls the overall gain of the receiver. Set this parameter to 50.

iii. Baseband sample rate (Hz): controls the sample rate of data delivered to the PC over the USB bus. This rate corresponds to the instantaneous bandwidth of the receiver. Set this parameter to 2e6.

iv. Samples per frame: specifies the length of each data frame that will be processed by Simulink. The choice 3660 should work well for this example.

NOTE: Once these settings are configured, these are copied over to the PLUTO device when you click the Run button below 'defining' the operations performed by the 'programmable' radio hardware/platform i.e., an SDR.

IV. Begin execution by clicking Run (the play button). After a few moments you should see a plot in the Spectrum Analyser window similar to Figure 3. This plot is the real-time spectrum of the 2 MHz wide RF band centred at 92.0 MHz. On the Frequency axis, the 0 MHz location corresponds to the centre frequency of the SDR. Notice the strong peak on the plot at 0.5 MHz, which approximately corresponds to 92.5 MHz broadcast (Phoenix FM Community Radio Station)



V. By default, the spectrum analyser performs exponential averaging to reduce the noise in the observation - to see the original form you can alter the averaging specifications under View → Spectrum Settings → Trace (to running, and set the averaging to 1). You can also observe running average noise reduction by using a setting other than 1.

VI. Experiment with the configuration of the PlutoSDR receiver block.

- What happens to the spectrum plot when you adjust the Center frequency parameter?
- What happens when you adjust the Gain parameter?
- What happens when you adjust the Baseband sample rate parameter? **Hint**: Be sure the Full Frequency span box is checked in the Spectrum Settings panel of the Spectrum Analyser.

4 Other wireless signals

Experiment with the Spectrum Analyser by viewing some of the many interesting signals that are literally surrounding you! Numerous real-world communication systems use frequencies within the operating range of the SDR hardware, such as

- FM Broadcast Radio (88-108 MHz)
- Personal transmitters such as car key fobs (315 MHz)
- Telecom systems (various bands between 700-5000 MHz)
- GPS (1227 MHz and 1575 MHz)
- WiFi (2400 MHz)
- Bluetooth (2400 MHz)

There are many more including amateur radio signals (both licensed and non-licensed), aircraft broadcasts and weather radars. A simple web-search can often tell you the operating frequency of RF systems (a list of Ireland's FM radio stations can be seen here)

Submission

Todo: Identify a real-world radio frequency signal of your choosing and investigate it! You can choose any signal you wish, so long as its operating frequency is within the range of your SDR receiver. Feel free to consult with your instructor if you have doubts whether your preferred signal is appropriate for this exercise.

Submit the following for your observation as a short report via blackboard:

- centre frequency of the signal
- screen capture of the spectrum analyser window showing the signal
- estimate of the signal bandwidth
- Describe any characteristics of the signal based on your research of the signal/system, or based on your observations of its spectrum. It is OK if you don’t fully understand the details at this stage.
- Observations from sub-section 3.1.VI

Acknowledgement: Labs adapted from C. Prust, Introductory Communication Systems Course using SDR, 2022.

An aside:

While Simulink offers a GUI-based method to build the intelligence, the same functionality could also be achieved directly via MATLAB scripts. Here is an example model for the spectrum analyser we investigated today directly through MATLAB. Note that MATLAB will be configured as the Industrial I/O (IIO) client using system objects *comm.SDRRxPluto* and *comm.SDRTxPluto* for the receiver and transmitter respectively.

Instantiating system objects:

```
>> rx = sdrrx('Pluto');
>> tx = sdrtx('Pluto');
```

The above sets up the objects, which can then be displayed or configured. Below are the default values you would see when an rx object is initialised.

```
>> rx
rx =
  comm.SDRRxPluto with properties:
   Main
                DeviceName: 'Pluto'
                   RadioID: 'usb:0'
           CenterFrequency: 2.4000e+09
                GainSource: 'AGC Slow Attack'
            ChannelMapping: 1
        BasebandSampleRate: 1000000
            OutputDataType: 'int16'
           SamplesPerFrame: 20000
           EnableBurstMode: false
    ShowAdvancedProperties: false
  Show all properties
```

The properties can be configured similar to how we did with the Simulink model. The specifications of these properties are as below:

- **RadioID:** Related to interface number of connected device, either via USB or Ethernet/ IP address
- **CenterFrequency**: Operating center frequency in Hz
- **BasebandSampleRate**: Baseband sampling rate in Hz
- **GainSource**: Gain specification of the receiver
  - Manual, AGC Slow Attack (default), AGC Fast Attack
- **ChannelMapping**: set to 1 (for Pluto device)
- **OutputDataType**: set to inherent MATLAB datatype: 16-bit complex integers
- **SamplesPerFrame**: Specifies the buffer sizing for MATLAB to Pluto interface

To replicate our spectrum analyser, we can use the below snippet. Note that the plot would look slightly different from our live data based Simulink model.

```
%Instantiate system object
rx = sdrrx('Pluto');
tx = sdrtx('Pluto');

% Configure the RX properties using the rx object
rx.CenterFrequency = 92e6;
rx.GainSource = 'Manual';
rx.Gain = 50;
rx.BasebandSampleRate = 2e6;
rx.SamplesPerFrame = 3660;

% Collect data from the rx interface
frameSize = 3660;
framesToCollect = 10e3;
data = zeros(frameSize, framesToCollect);
for frame = 1:framesToCollect
    [d,valid,of] = rx();
    if ~valid
        warning('Data invalid');
    else
        data(:,frame) = d;
    end
end

% Process collected data using the spectrum-analyzer
sa1 = dsp.SpectrumAnalyzer;
%Configure the spectrum analyzer with similar parameters
sa1.SampleRate = 2e6;
sa1.InputDomain = 'Time';
sa1.ViewType = 'Spectrum';
sa1.Method = 'Filter bank';
sa1.NumTapsPerBand = 12;
for frame = 1:framesToCollect
    sa1(data(:,frame)); %process the frame
end
```